Graph Neural Networks for fast emulation of nuclear interaction models

G4 Collaboration Meeting 2023 - Hokkaido University, Sapporo

L. Arsini[1,2], B. Caccia[3], A. Ciardiello[1], M. Colonna[4], S. Giagu[1,2], C. Mancini Terracciano[1,2]

[1]Department of Physics,Sapienza University of Rome, Rome, Italy. [2]INFN, Section of Rome, Rome, Italy. [3]Istituto Superiore di Sanità, Rome, Italy. [4]INFN, Section of LNS

26/09/2023

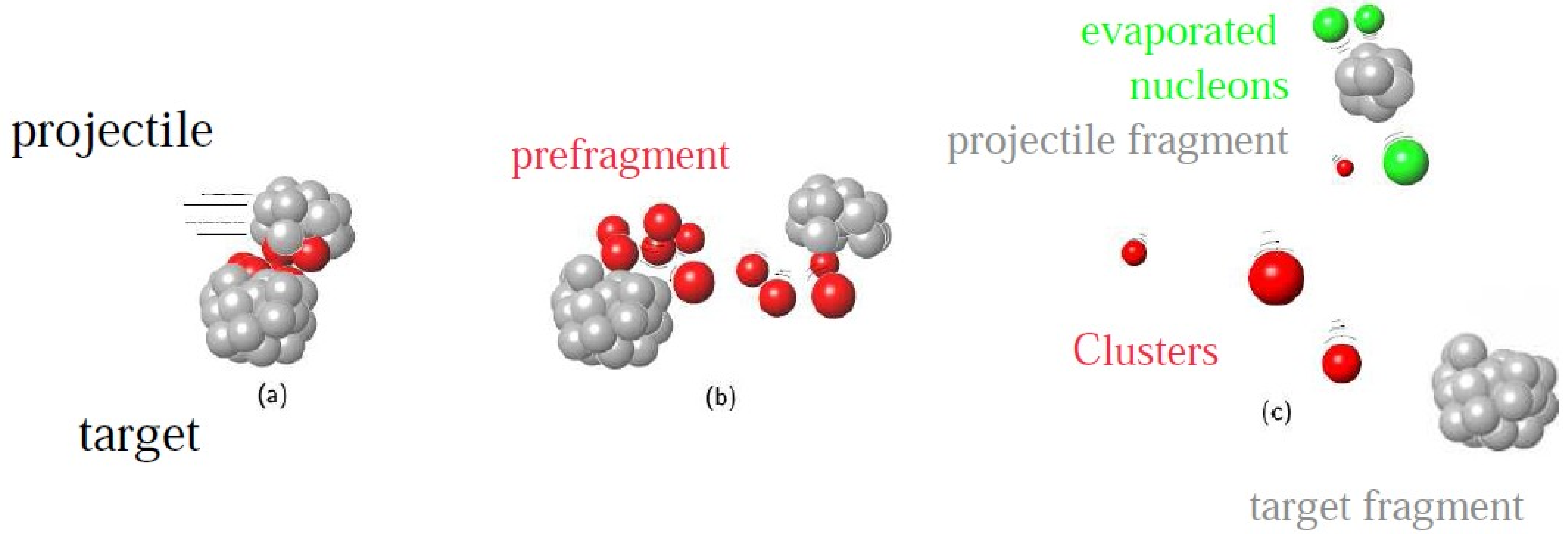
Nuclear interaction models

In Geant4

- Nuclear interaction models are **slow**
- In particular the most sophisticated ones e.g. **QMD**

Trade off between computing **time** and **precision**

Use simpler models



Problems in Geant4 below 100 MeV/u

No dedicated model to nuclear interaction **below 100 MeV/u** in Geant4

Many papers showed discrepancies:

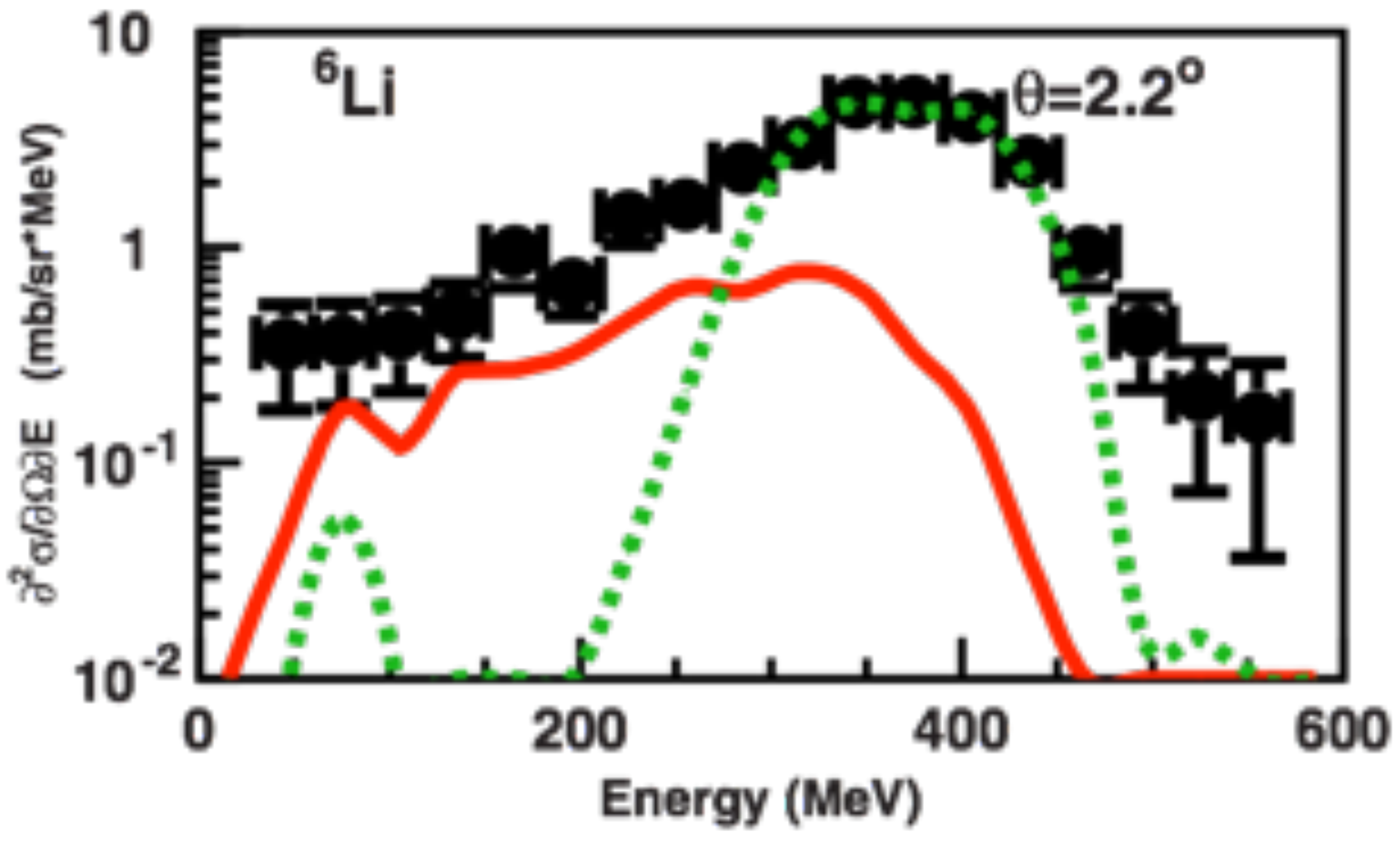
Braunn et al. : one order of magnitude in 12C fragmentation at 95 MeV/u on thick PMMA target

De Napoli et al. : angular distribution of the secondaries emitted in the interaction of 62 MeV/u 12C on thin carbon target

Dudouet et al. : similar results with a 95 MeV/u 12C beam on H, C, O, Al and Ti targets

- **Exp. data**
- **G4-BIC**
- **G4-QMD**

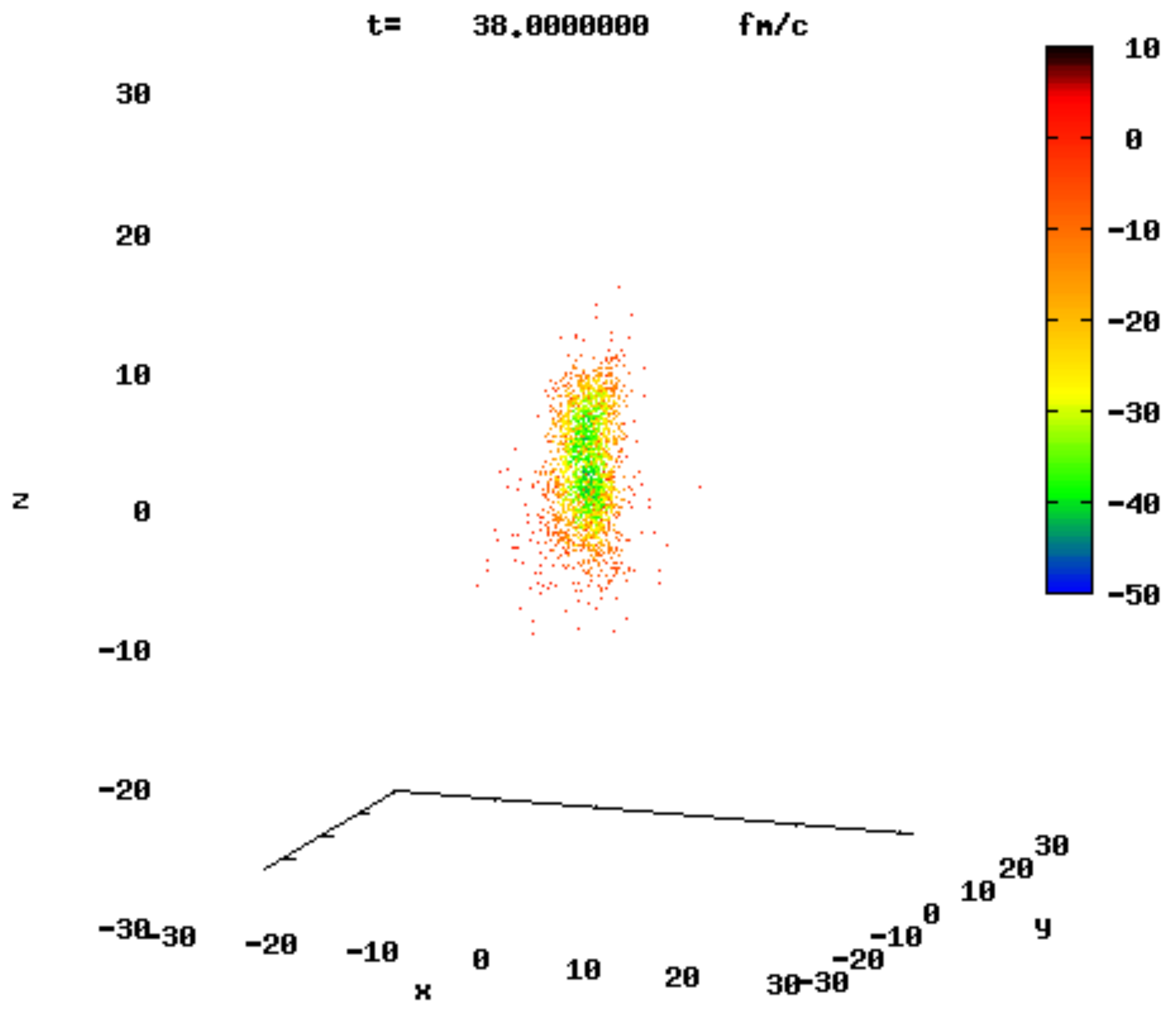
[Plot from De Napoli et al. Phys. Med. Biol., vol. 57, no. 22, pp. 7651–7671, Nov. 2012]



Cross section of the ^{6}Li production at 2.2 degree in a ^{12}C on ^{nat}C reaction at 62 MeV/u.

BLOB (Boltzmann-Lagevein One Body)

- Test-particle approach
- Self-consistent **mean field** + collisions
- Probability to find a nucleon in the phase space



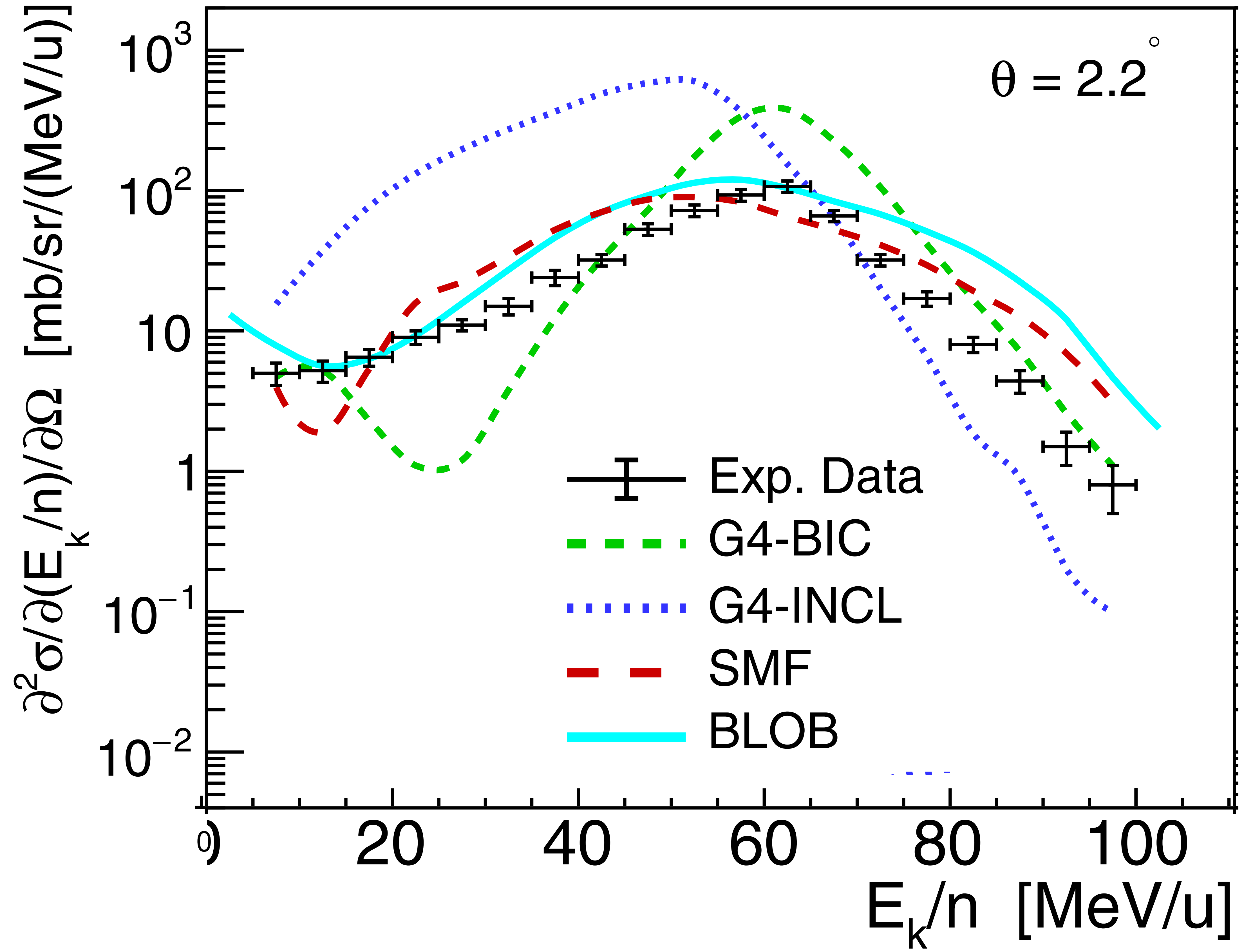
BLOB (Boltzmann-Lagevein One Body)

We interfaced BLOB with Geant4 and its de-excitation model

[C. Mancini-Terracciano et al. Preliminary results coupling “Stochastic Mean Field” and “Boltzmann-Langevin One Body” models with Geant4. In: Physica Medica 67 (2019), pp. 116–122. doi: 10.1016/j.ejmp.2019.10.026.]

Accurate **Slow**

Order of minutes per interaction!



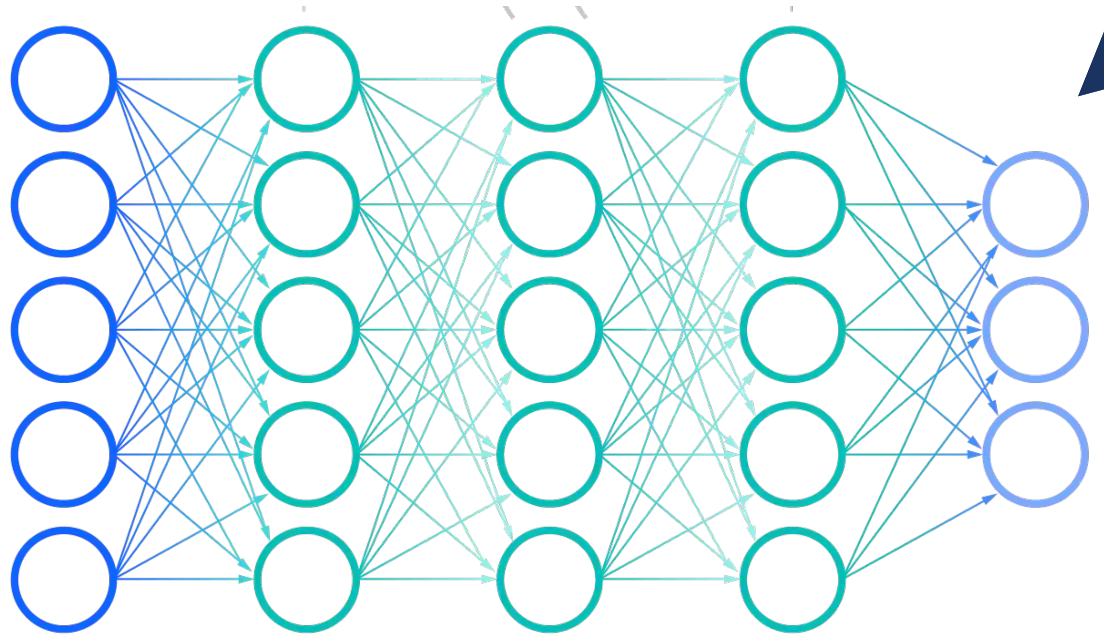
Deep Learning to emulate NIMs

$$f(X) = Y$$

X

Y

$$\tilde{f}_w(X) = Y$$



Neural Network

Reproduce a model output

Negligible running time

Complex Physics Simulations

Sanchez-Gonzalez, Alvaro, et al. "Learning to simulate complex physics with graph networks." *International Conference on Machine Learning*. PMLR, 2020.

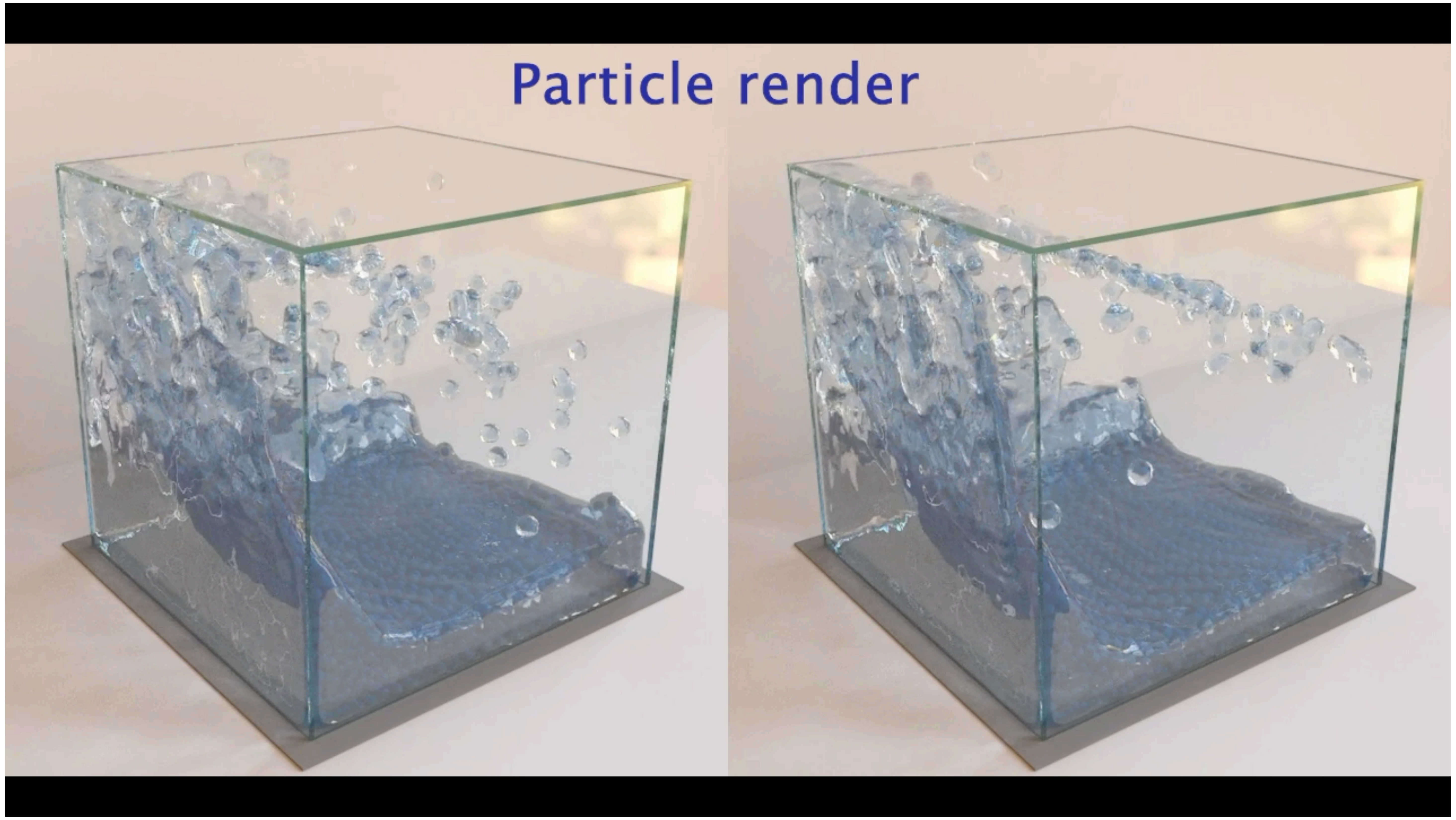
https://arxiv.org/abs/2002.09405

Github

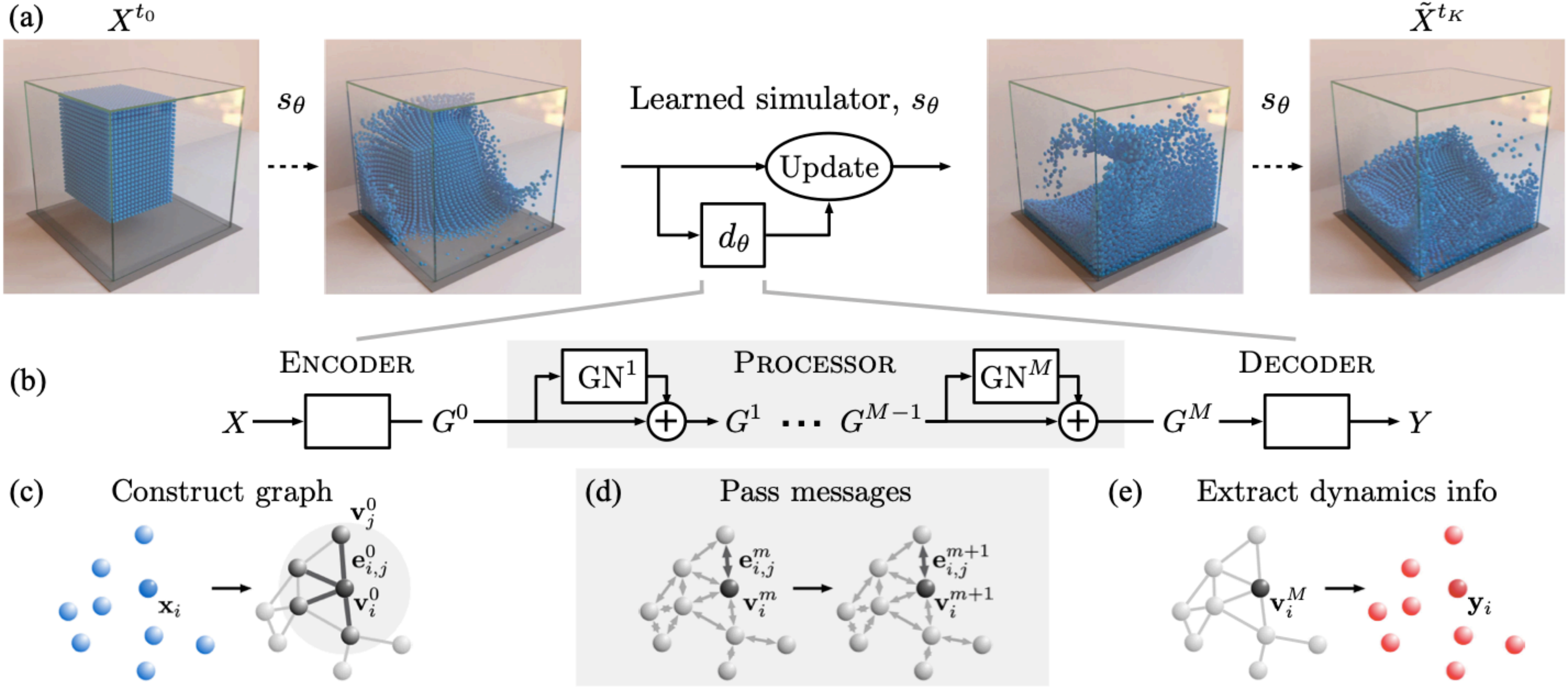
github.com/deepmind/deepmind-research/tree/master/learning to simulate.

Videos

https://sites.google.com/view/learning-to-simulate

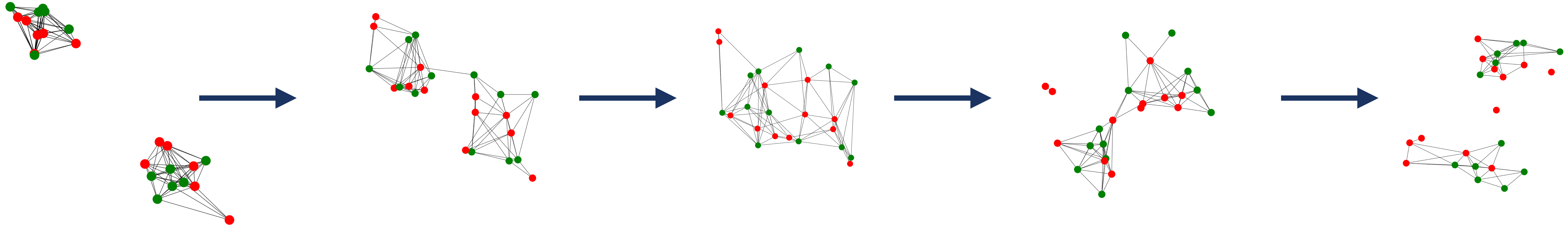


Graph Network-based Simulators (GNS)



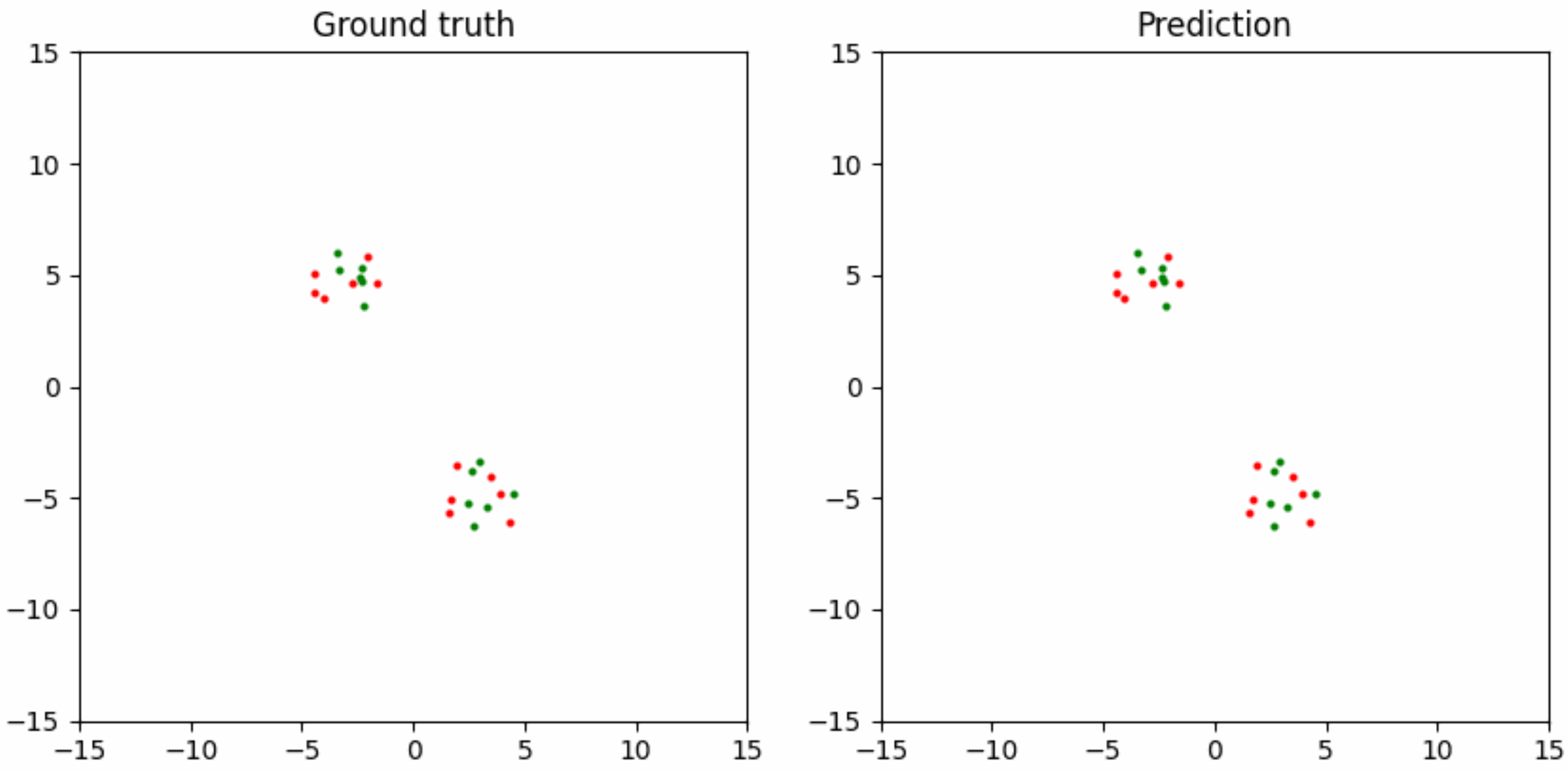
GNN approach

Emulating the dynamics of **QMD** in ^{12}**C on** ^{12}**C** reaction at **12 MeV/u**



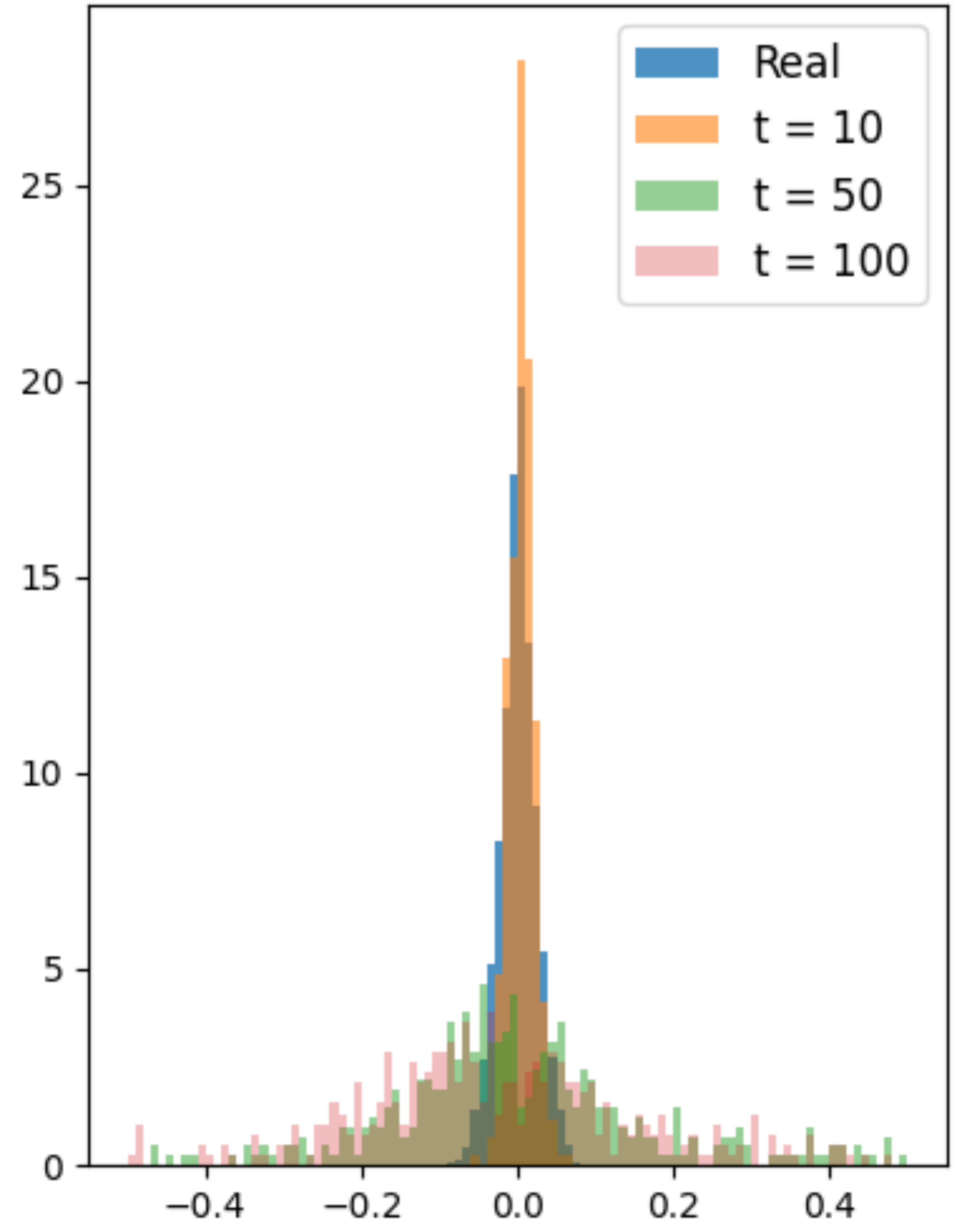
Each nucleon is a node of the graph

Visually satisfying results…



… which are not Physical

Momentum on z axis



In the **best case scenario**:

> Quantities are conserved on average
>
> Variance explodes increasing time steps

Cannot be used to infer physical quantities at the end of the reaction

Differences in the Physics

QMD, BLOB

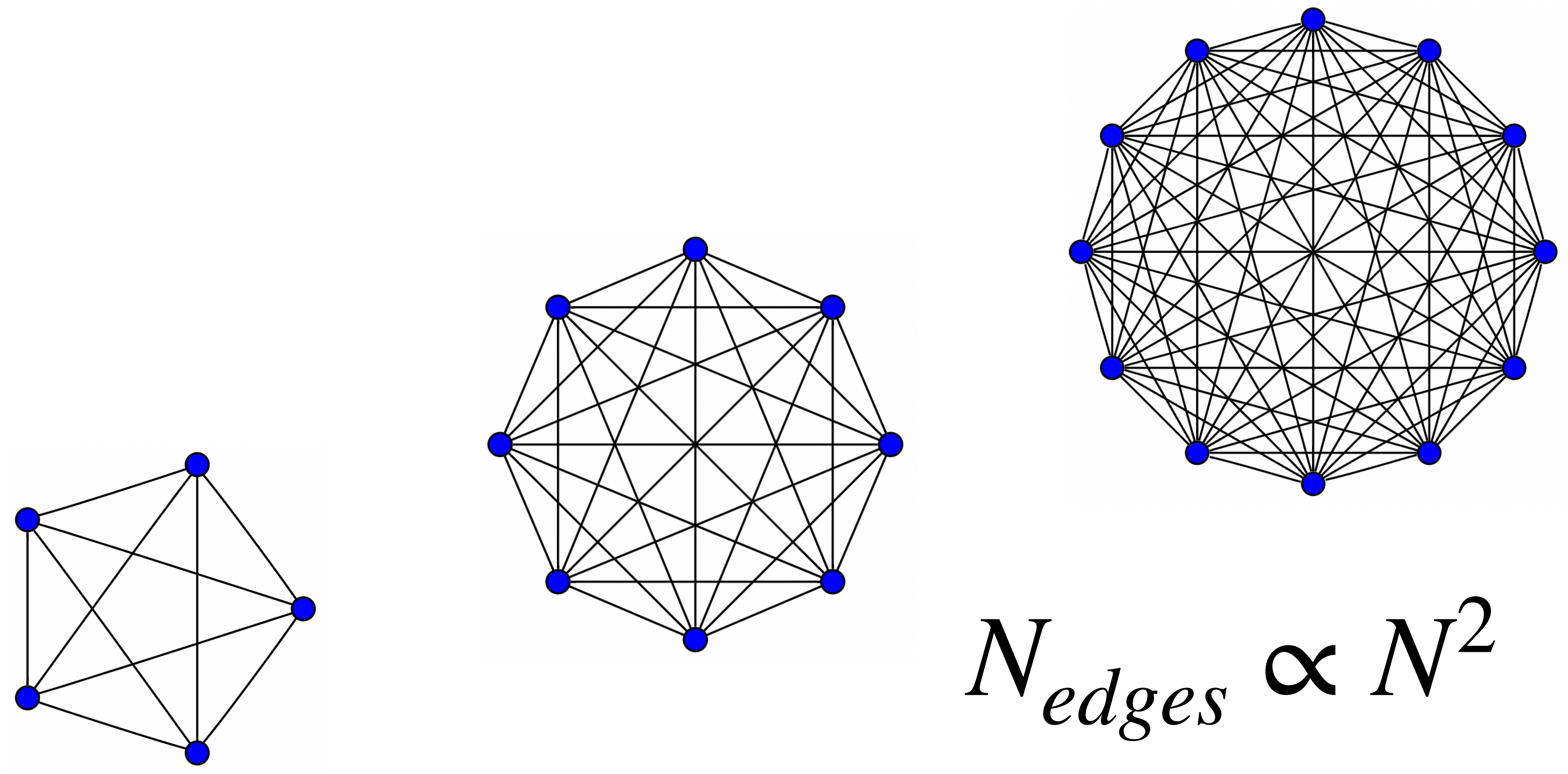
Long range interactions
Collisions

Liquid simulations

Short range interactions
Gravity

→ Building **fully connected** graphs

Feasible only for
limited number of nodes
(**QMD**)



$$N_{edges} \propto N^2$$

Proposed approach

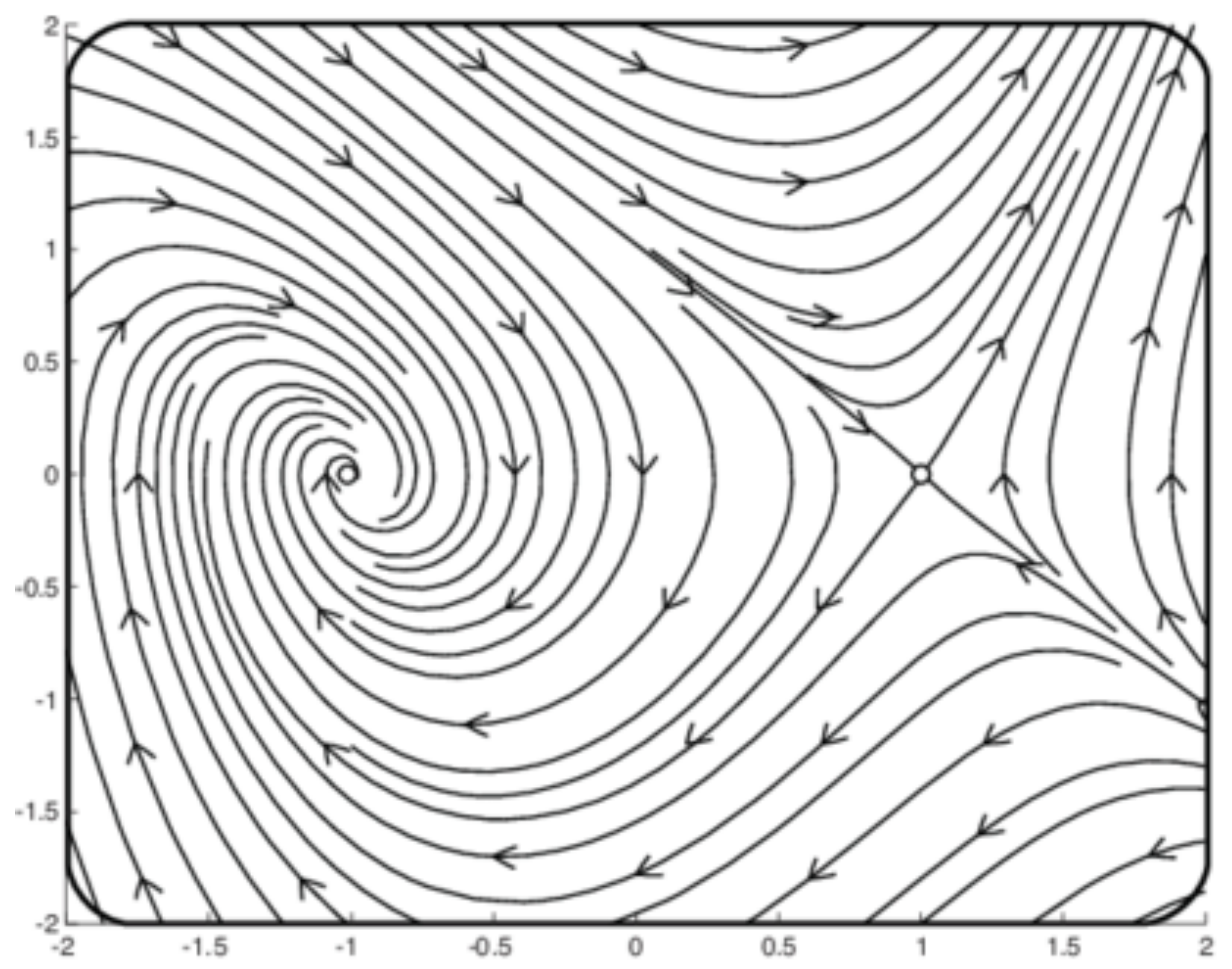
QMD, BLOB

Long range interactions
Collisions

Liquid simulations

Short range interactions
Gravity

Emulating the **Potential**
acting on each node



Why the Potential

Three good reasons

Get more **control** on the Physics

Speed up: Potential computation is the Bottleneck

Improving Mean Field

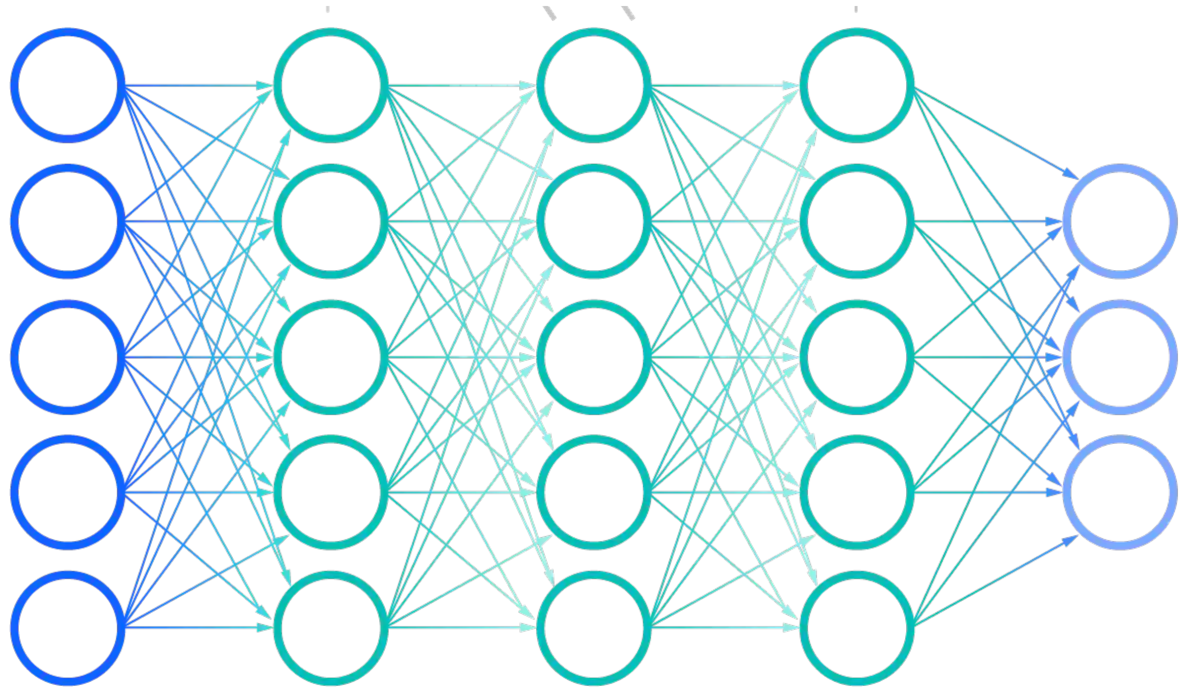
Why the Potential

Get more control on the Physics

No blackbox AI solution

DL computes a **known**, but complex function

$$\{q, p\} \longrightarrow$$



$$\longrightarrow V_i$$

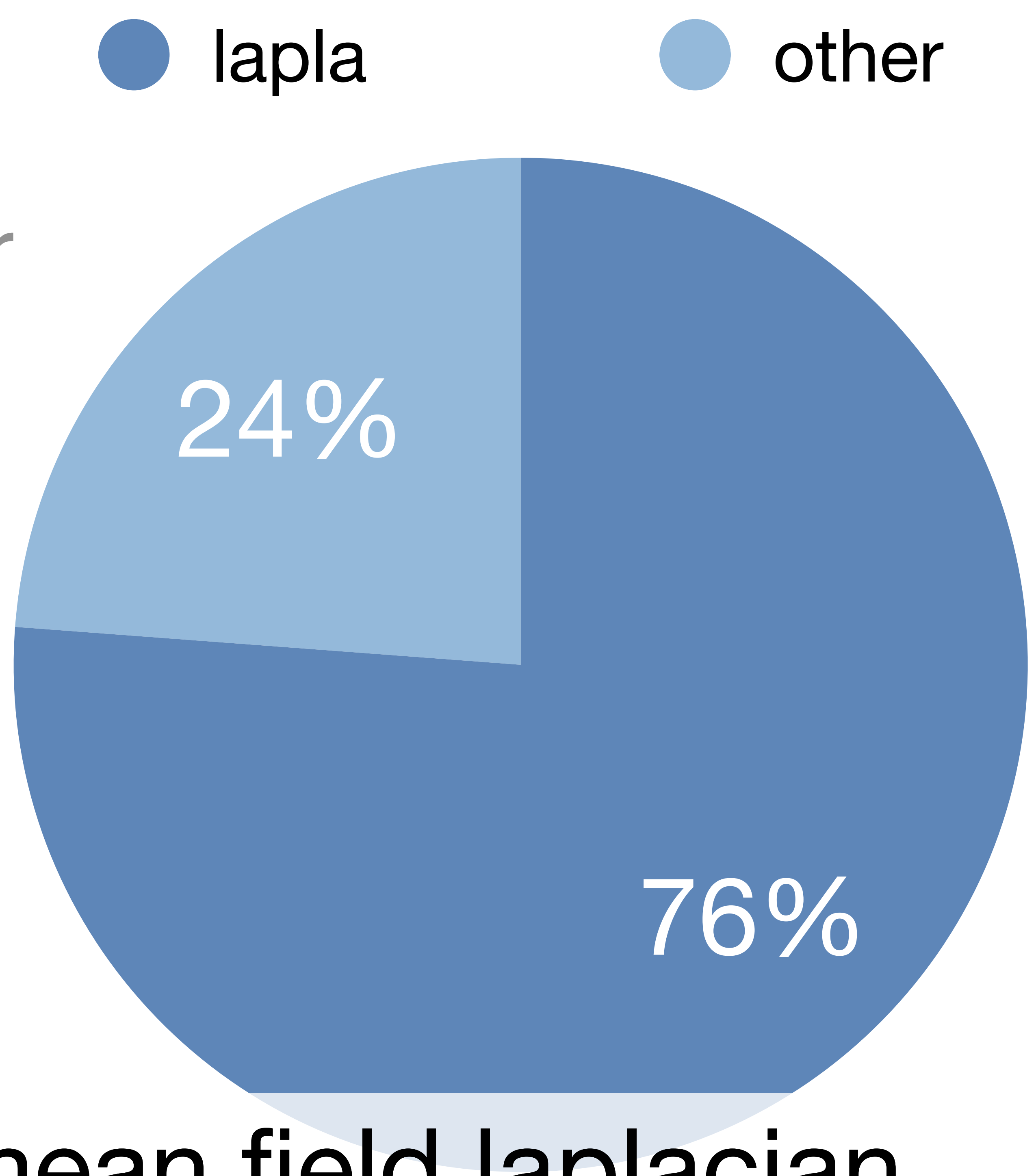
Enforce physical **conservation laws** in the model

Why the Potential

It is the Bottleneck

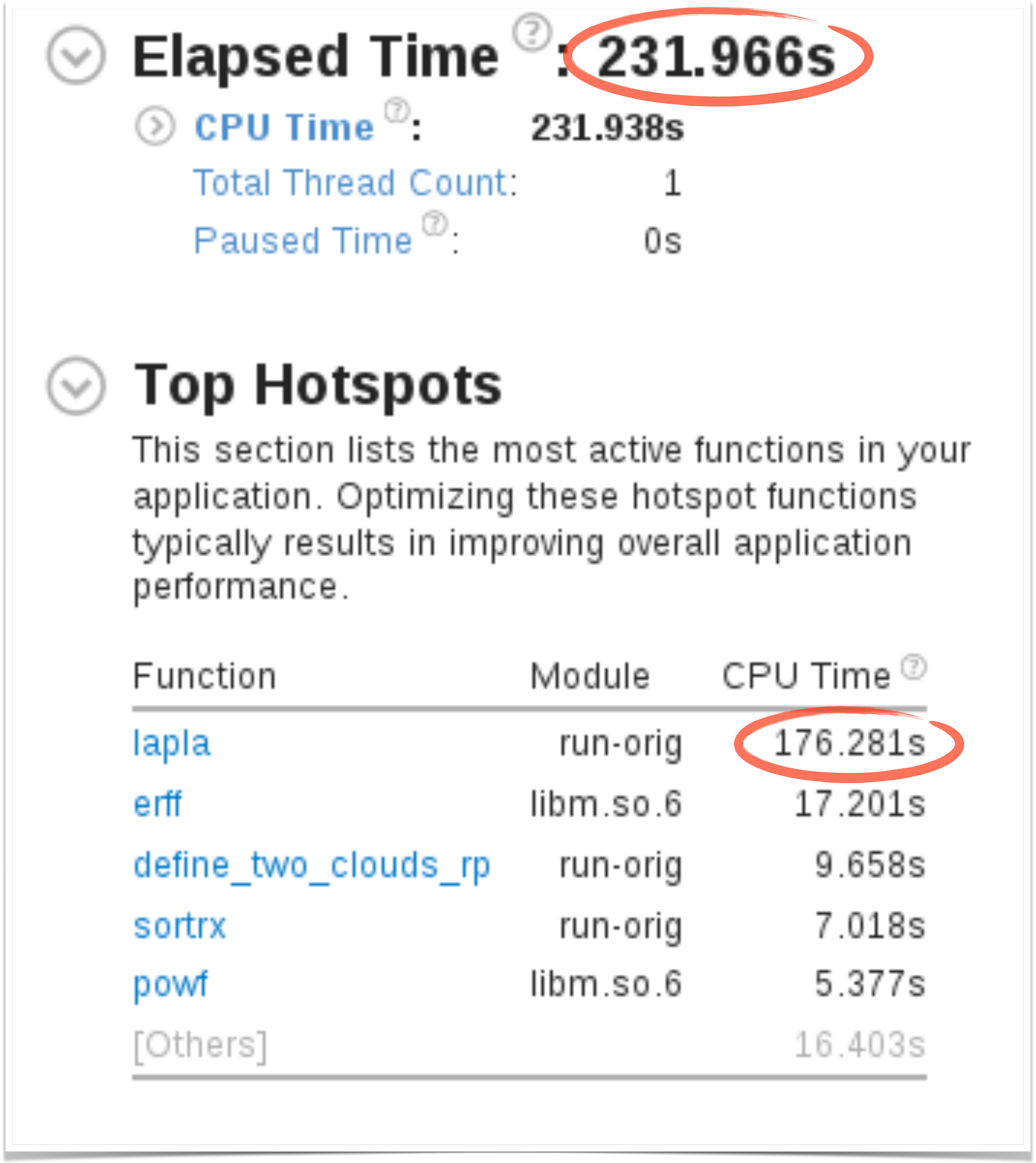
Profiling BLOB

with Intel VTune Amplifier



~ 4 mins per interaction

3 mins: computing mean field laplacian



Elapsed Time: 231.966s

CPU Time: 231.938s

Total Thread Count: 1

Paused Time: 0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time
lapla	run-orig	176.281s
erff	libm.so.6	17.201s
define_two_clouds_rp	run-orig	9.658s
sortrx	run-orig	7.018s
powf	libm.so.6	5.377s
[Others]		16.403s

Why the Potential

Improving Mean Field

Learn **any potential** given particle coordinates

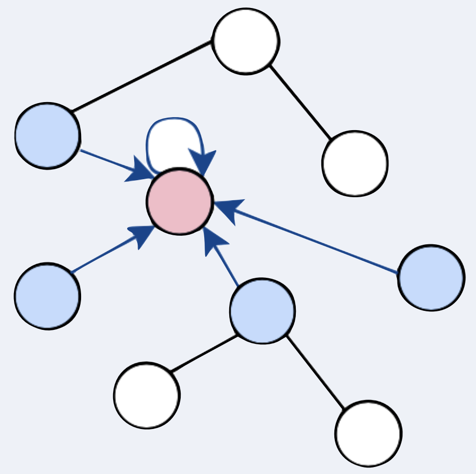
No more time and complexity **overload**

Improve Mean Field approximation

Once you've learned the Potential

Full Deep Learning

V_i Accounts for long range interactions



GNNs predicts collisions

$$\{q^t, p^t, V^t\} \underbrace{}_{\text{FDL}} \{q^{t+1}, p^{t+1}\}$$

Hybrid Models

Get F_i differentiating V_i

Integrate the equation of motion with standard methods

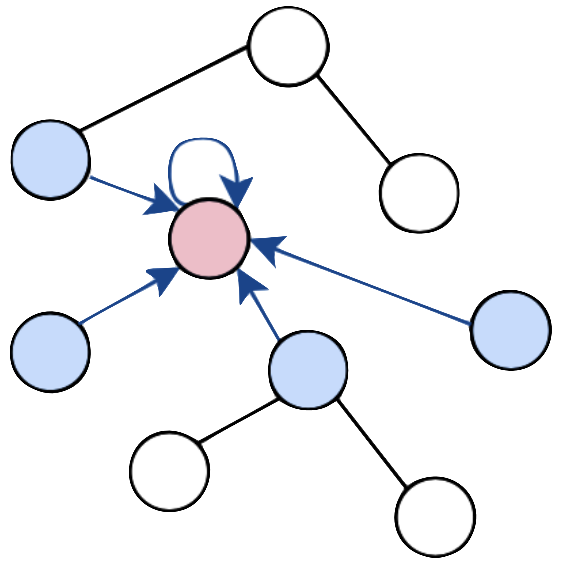
(Runge Kutta 4, ...)

Once you've learned the Potential

Full Deep Learning

V_i

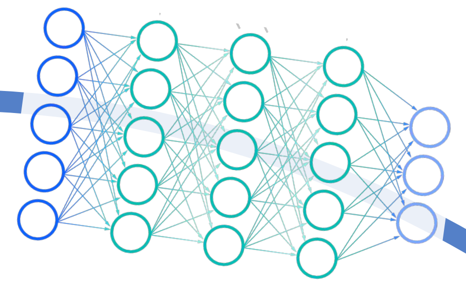
Accounts for long range interactions



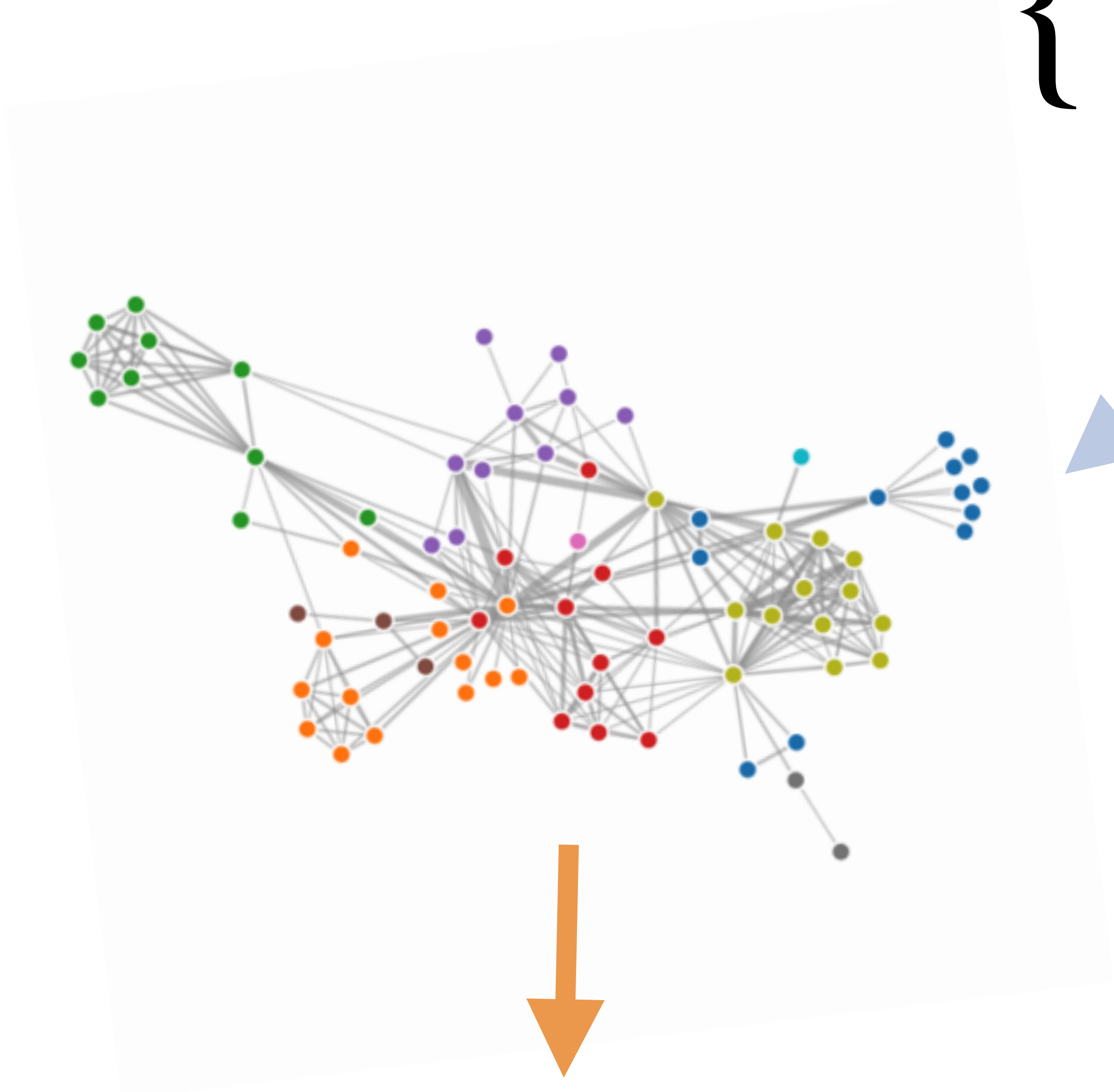
GNNs predicts collisions

No need for Fully Connected graphs

$\{q^t, p^t\}$



$\{V^t\}$



$\{q^{t+1}, p^{t+1}\}$

Once you've learned the Potential

Mixing Deep Learning with standard methods

Hybrid Models

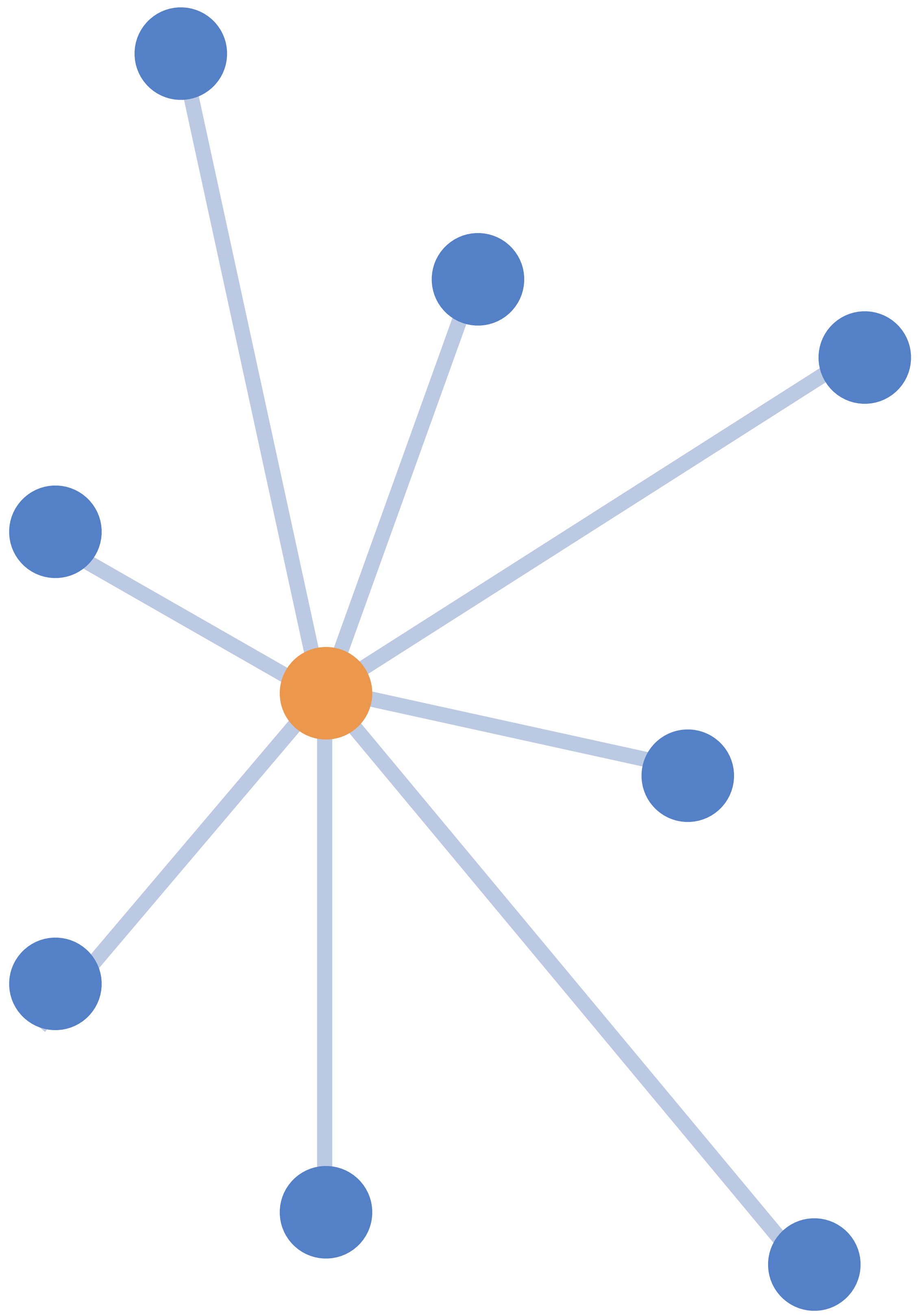
Get F_i **automatically differentiating** V_i → **Integrate** the equation of motion with standard methods

Export the DL model as a function → Integrate it into existing code, **replacing** potential computation

GetPotential() method in QMD

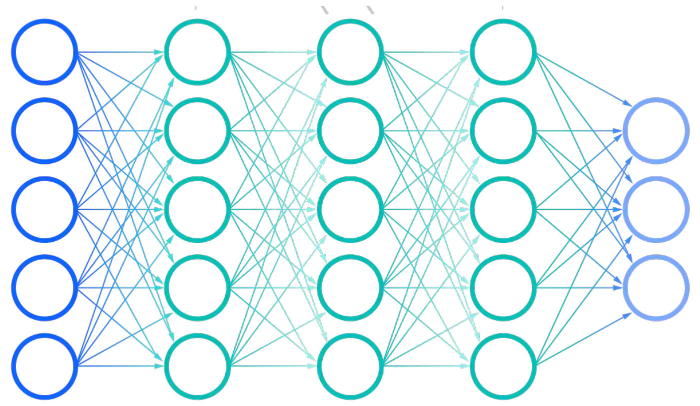
Learning the Potential: DL model

Particle-wise MLP for Potential Prediction



$$V_i = \sum V_{ij} = \sum f\left(q_i, q_j, p_i, p_j, c_i, c_j\right)$$

$=$



MLP

Embed particle exchange symmetry

Learning the Potential: Preliminary results

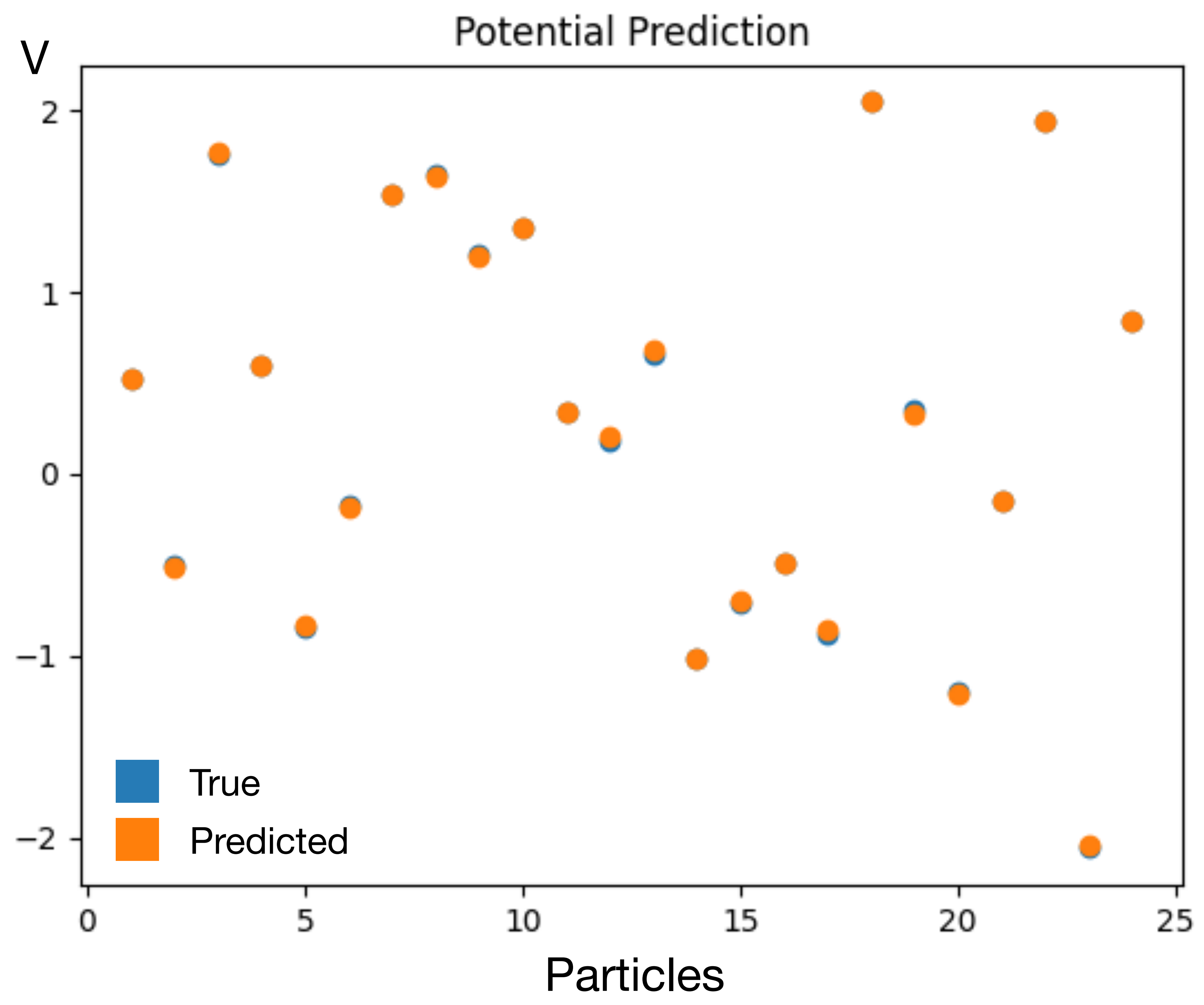
Particle-wise MLP for Potential Prediction

Model: 5 layers MLP + ReLu + LayerNorm

Data: 23k stories
10 events
24 particles : ~5 M examples

Training: ~3 days training on Nvidia V100

Results: Mean Absolute Error: 0.0155



Next Steps

Full Deep Learning approach to emulate QMD dynamics

Potential prediction on BLOB

Exporting the DL models in **ONNX** or using **Libtorch**

Releasing the code for **Geant4** integration (as an example?)

Thank you for your attention!

- Nuclear interaction models in Geant4:
  - Sophisticated models are slow
  - No dedicated model under 100 MeV/u
- Deep Learning approach for model emulation
  - State-of-the-art approach fails on QMD dynamics
  - Potential prediction with Deep Learning
  - **Full** Deep Learning or **Hybrid** models

SAPIENZA UNIVERSITÀ DI ROMA
ISTITUTO SUPERIORE DI SANITÀ
INFN

Backup slides

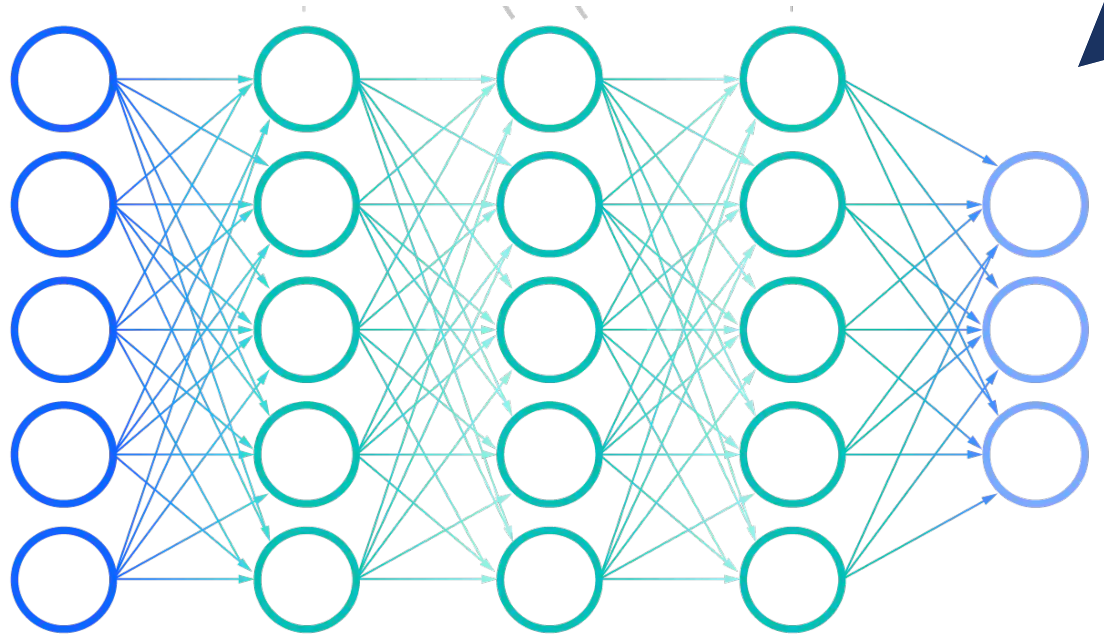
Deep Learning framework

X

$$f(X) = Y$$

Y

$$\tilde{f}_w(X) = Y$$



Neural Network

Known gradients

$\nabla_w \tilde{f}_w(X)$ wrt **parameters**

$\nabla_X \tilde{f}_w(X)$ wrt **input**

Differentiability

Training a Neural Network

Minimization of a *Loss Function*

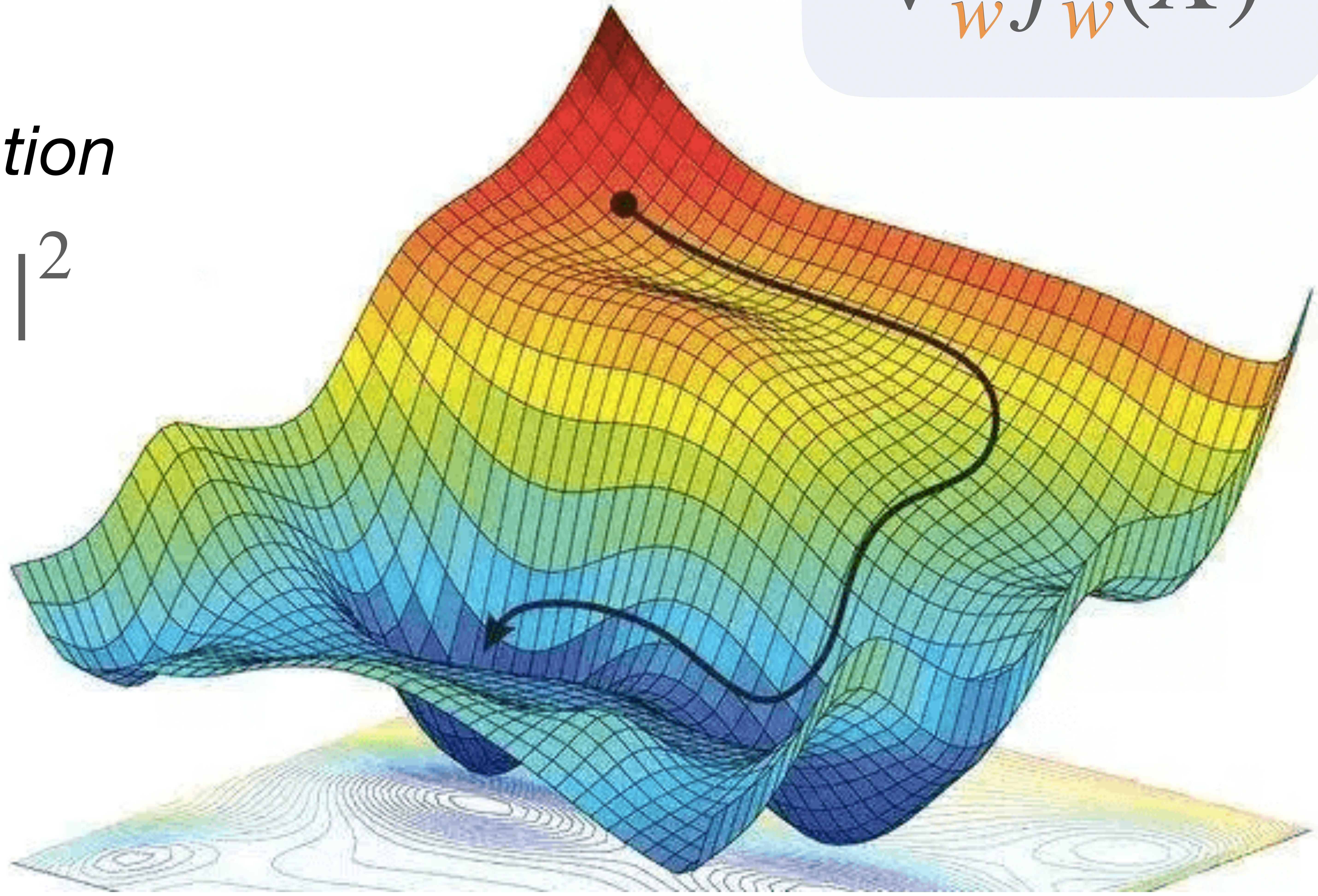
$$\mathscr{L}_w = |f(X) - \tilde{f}_w(X)|^2$$

Gradient descent

↓

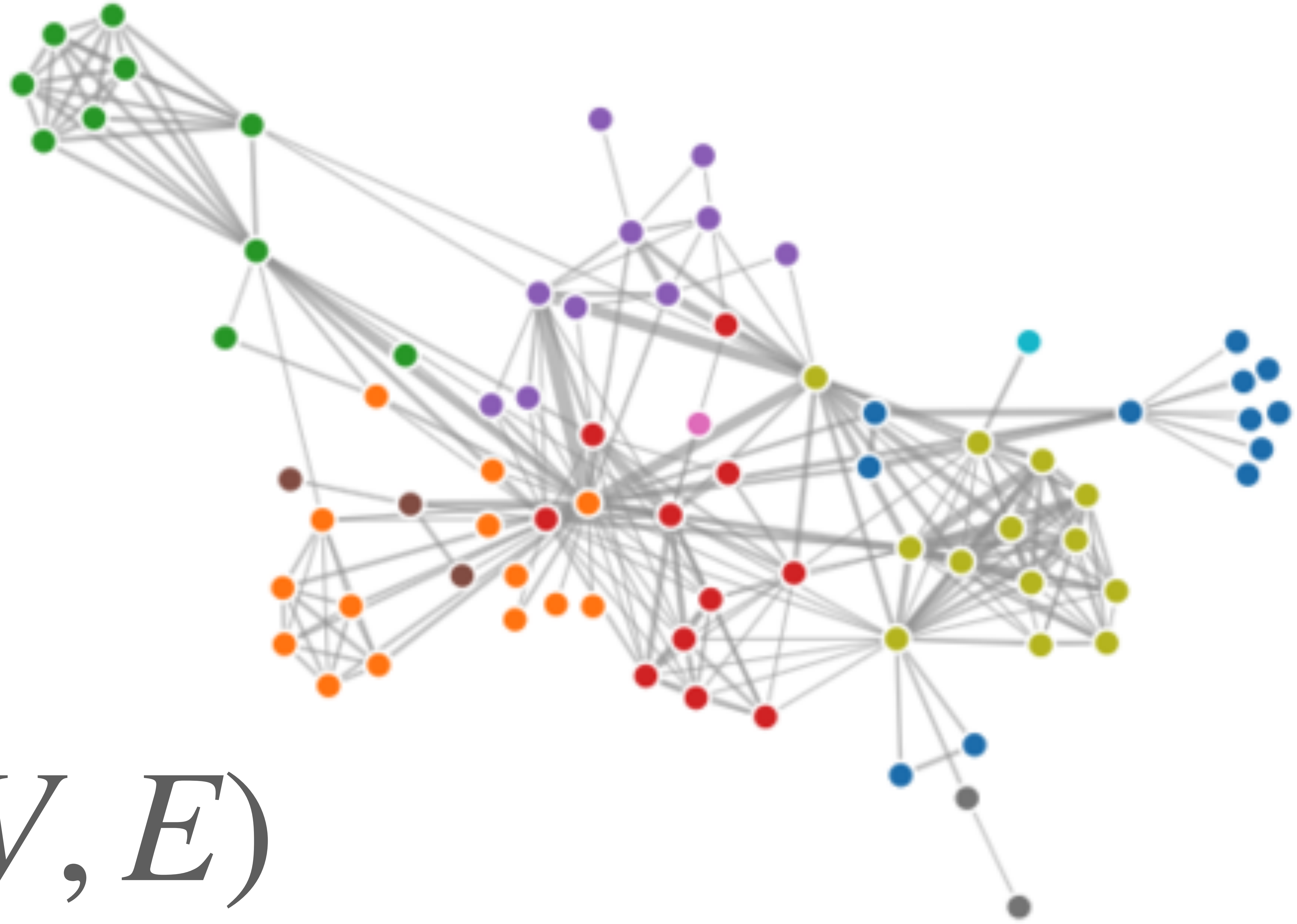
$$\min_w \mathscr{L}_w$$

$$\nabla_w \tilde{f}_w(X)$$

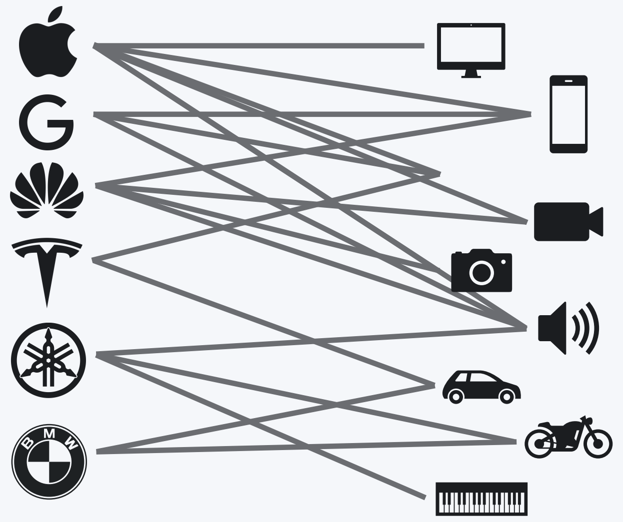


Graph Neural Networks

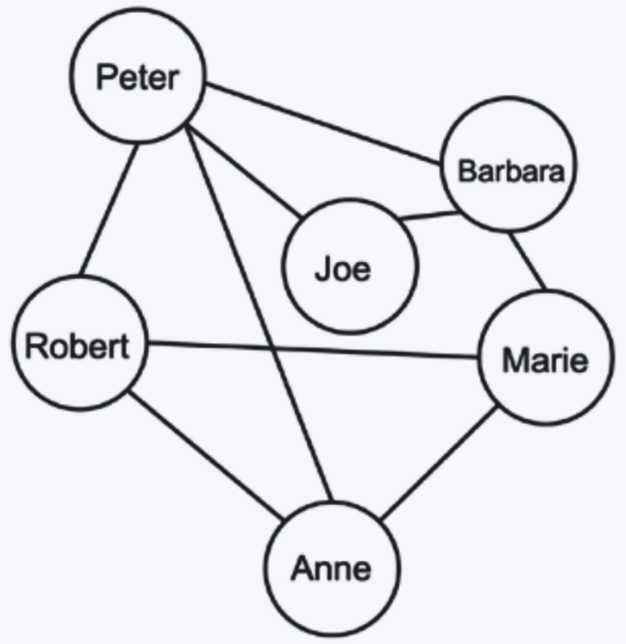
Deep Learning on Graphs



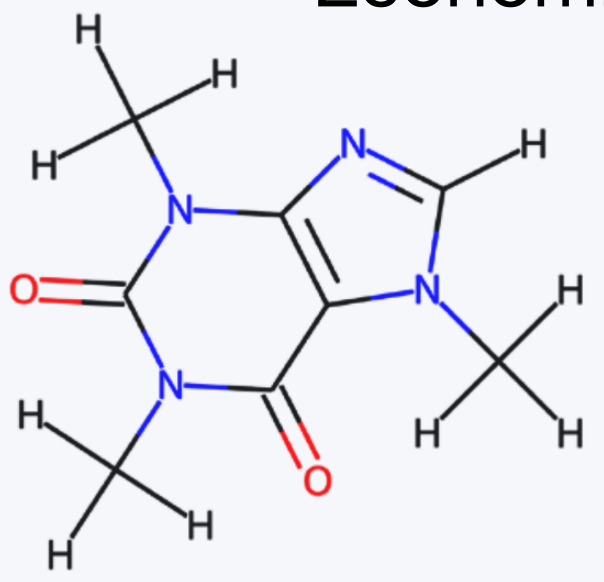
$$\mathcal{G}(V, E)$$



Economics

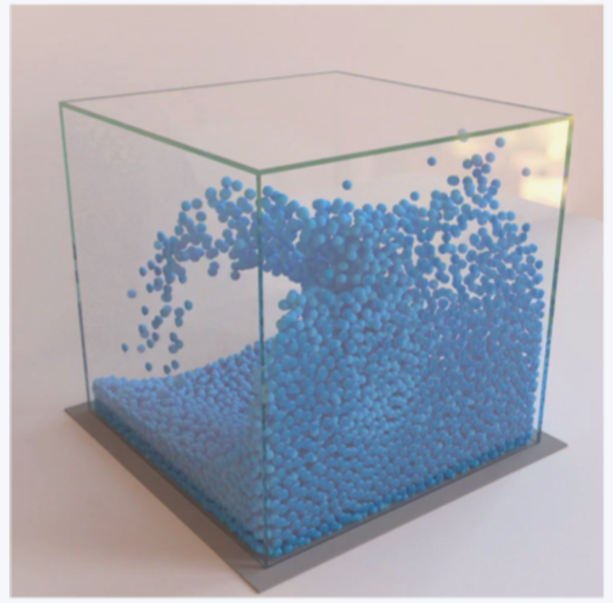


Social Networks



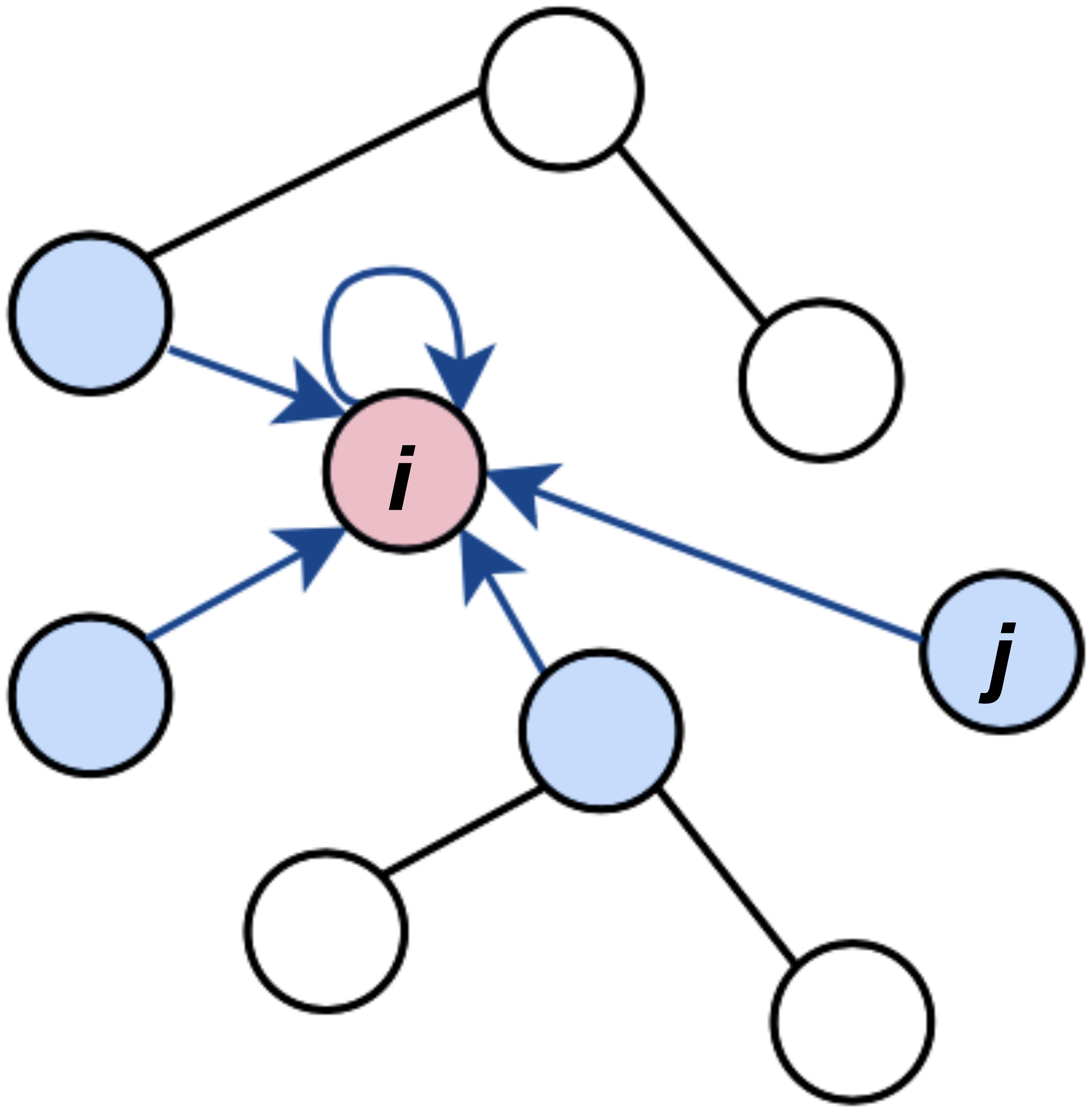
Molecules

Point Clouds



Physics Simulations

Message passing



Neighborhood information

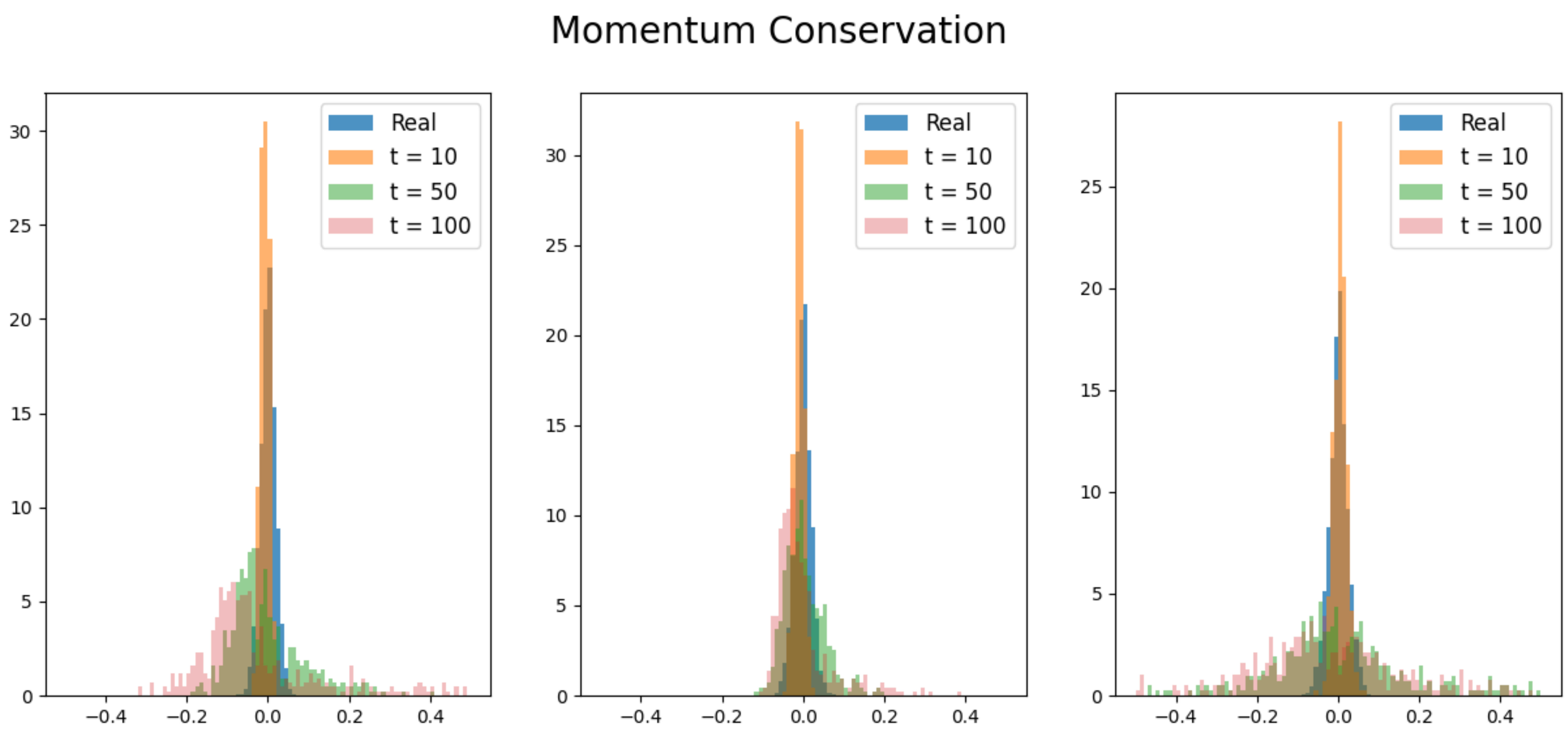
↓

Process — Learning on **nodes**

↓

Aggregate — Learning on **edges**

Momentum



Check Physical
conserved quantities

QMD conserves momentum on average

while

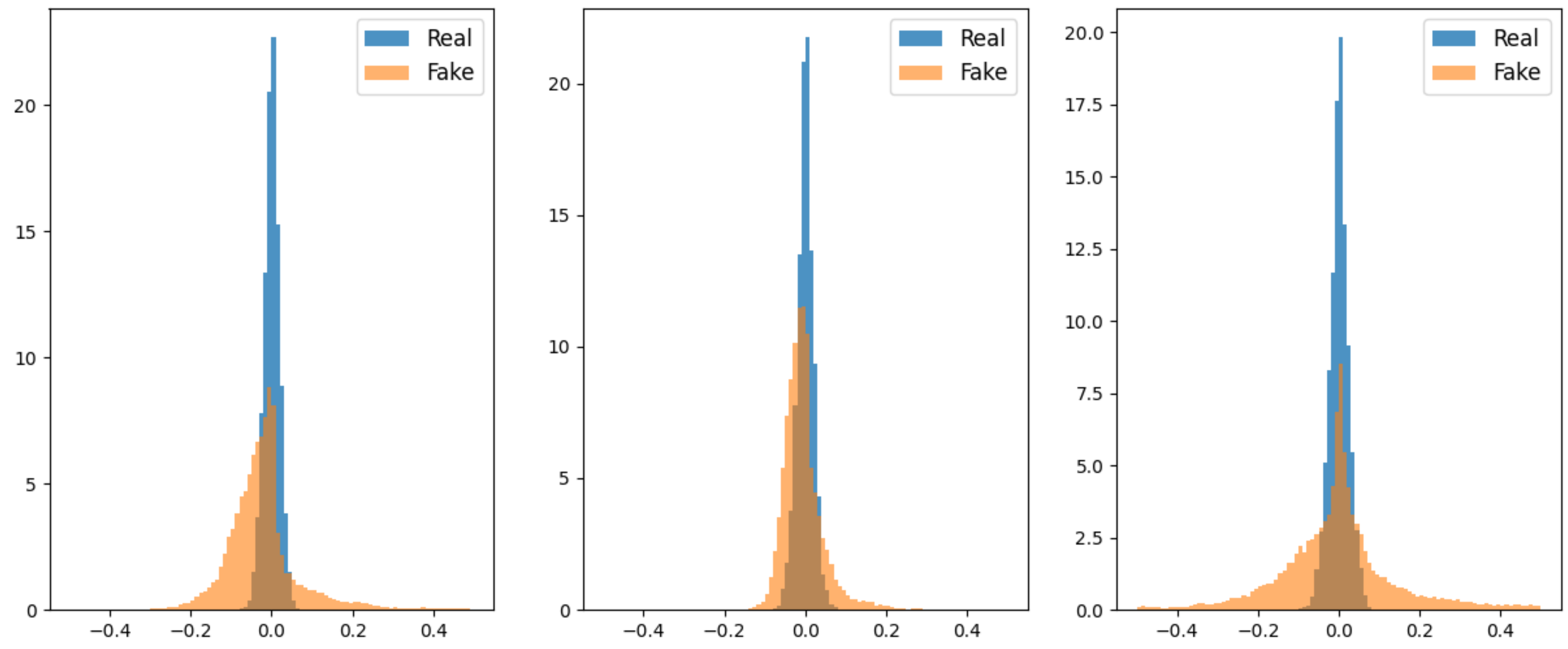
Graph-QMD:

- Similar mean
- Wider variance

Conservation fails at later times

Momentum

Momentum Conservation



Check Physical **conserved quantities**

QMD conserves momentum on average

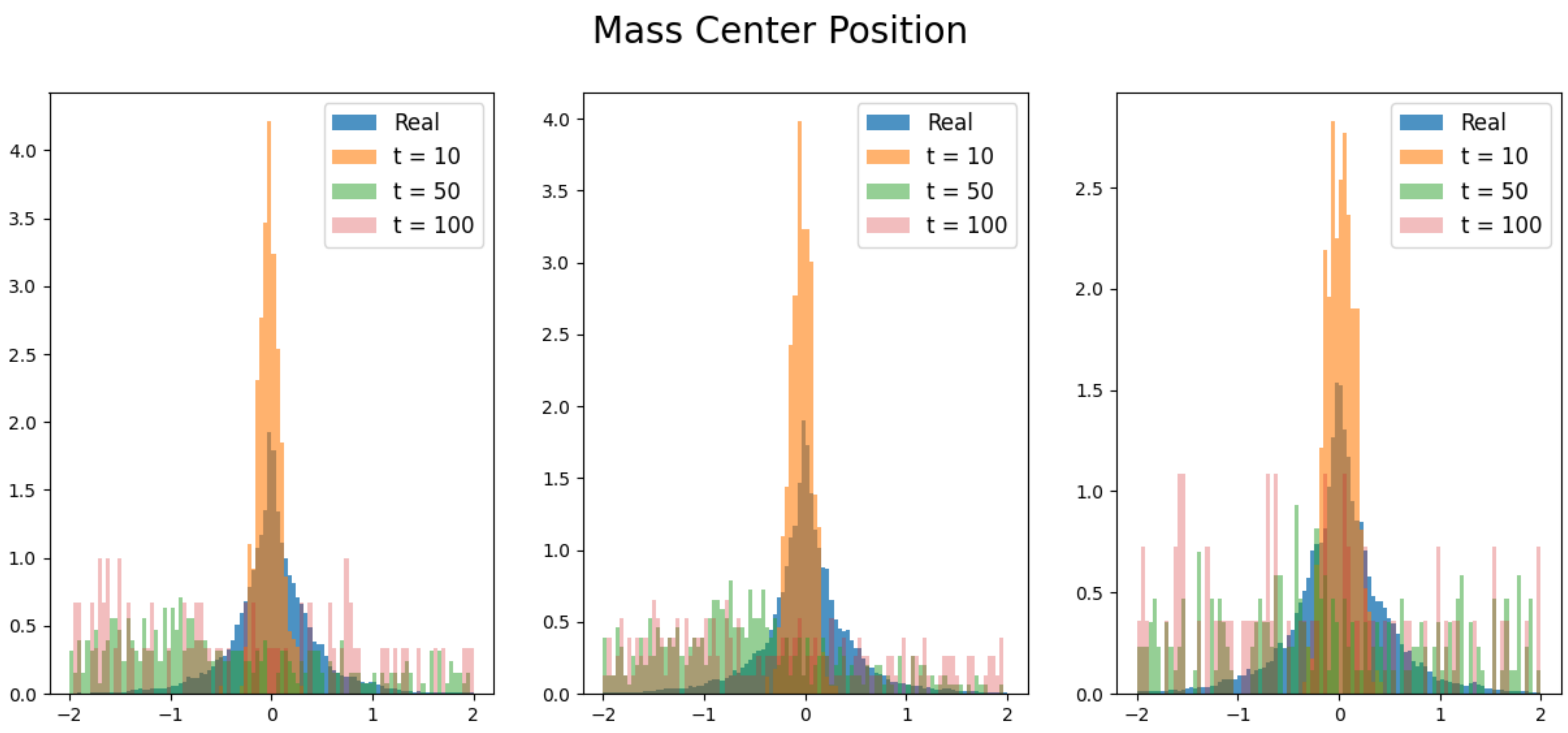
while

Graph-QMD:

- Similar mean
- Wider variance

Conservation fails at later times

Mass center position



Check Physical
conserved quantities

QMD conserves
mass center on average

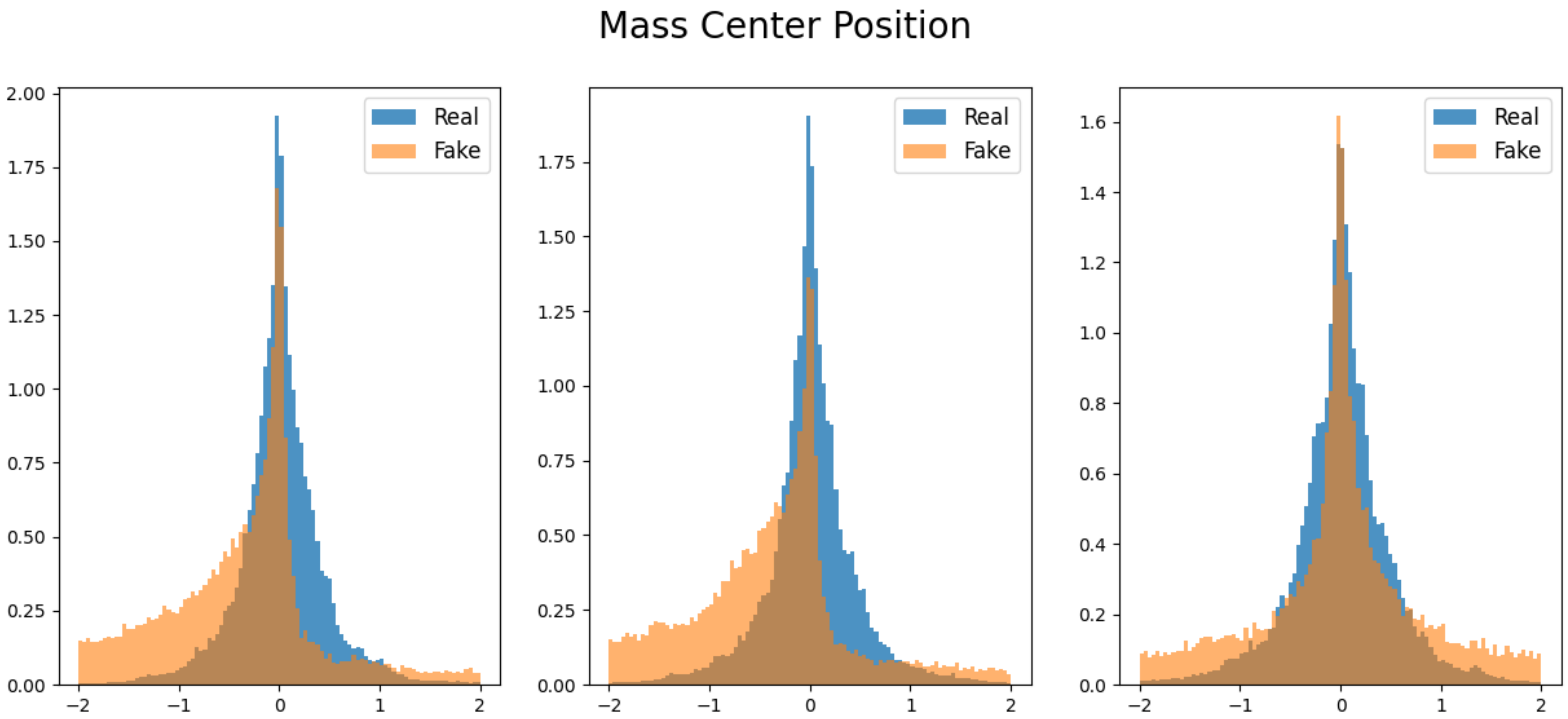
while

Graph-QMD:

- Similar mean
- Wider variance

Conservation fails at later times

Mass center position



Conservation fails at later times

Check Physical **conserved quantities**

QMD conserves mass center on average

while

Graph-QMD:

- Similar mean
- Wider variance